

DOCUMENT RESUME

ED 057 614

EM 009 465

TITLE Project Solo; Newsletter Number Twenty.
INSTITUTION Pittsburgh Univ., Pa. Dept. of Computer Science.
SPONS AGENCY National Science Foundation, Washington, D.C.
PUB DATE 20 Dec 71
NOTE 53p.; See also ED 053 566

EDRS PRICE MF-\$0.65 HC-\$3.29
DESCRIPTORS *Computer Assisted Instruction; *Computer Programs;
*Computer Science Education; *Mathematics
Instruction; Programing; *Secondary School
Mathematics

IDENTIFIERS *Project Solo

ABSTRACT

Three Project Solo modules are presented. They are designed to teach the concepts of elementary matrix operation, matrix multiplication, and finite-state automata. Together with the module on communication matrices from Newsletter #17 they form a well motivated but structured path to expertise in this area. (JY)

PROJECT SOLO

AN EXPERIMENT IN REGIONAL COMPUTING
FOR SECONDARY SCHOOL SYSTEMS

U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRO-
DUCED EXACTLY AS RECEIVED FROM
THE PERSON OR ORGANIZATION ORIG-
INATING IT. POINTS OF VIEW OR OPIN-
IONS STATED DO NOT NECESSARILY
REPRESENT OFFICIAL OFFICE OF EDU-
CATION POSITION OR POLICY



University of Pittsburgh • Department of Computer Science • Pittsburgh, Pennsylvania 15213

Newsletter No. 20

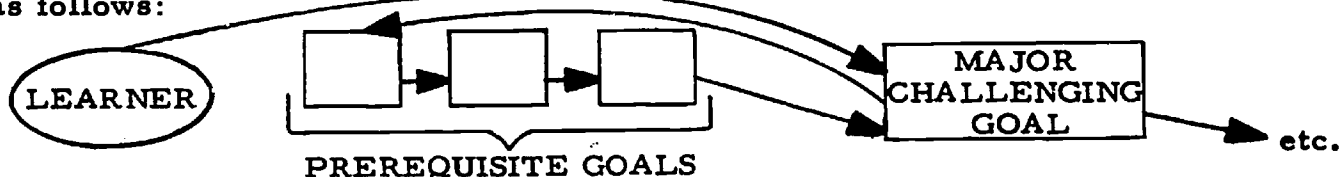
December 20, 1971

Learner Determined Goals and Structured Learning Sequences

There are a number of good reasons for questioning the suitability of pre-determined, linear sequences of educational objectives for learning. Newsletter No. 17 cited some of the questions that have been raised about the rigidity of such systems.

The real challenge seems to be one of combining the structure and guidance offered by a well-planned system with the freedom that the human learner needs in order to individualize his or her understanding of the subject under consideration.

One validated approach to resolving this dilemma is to set goals that are relatively few in number, but attractive and meaningful enough to invite learner selection of a sequence of pre-requisite goals. The process might be represented pictorially as follows:

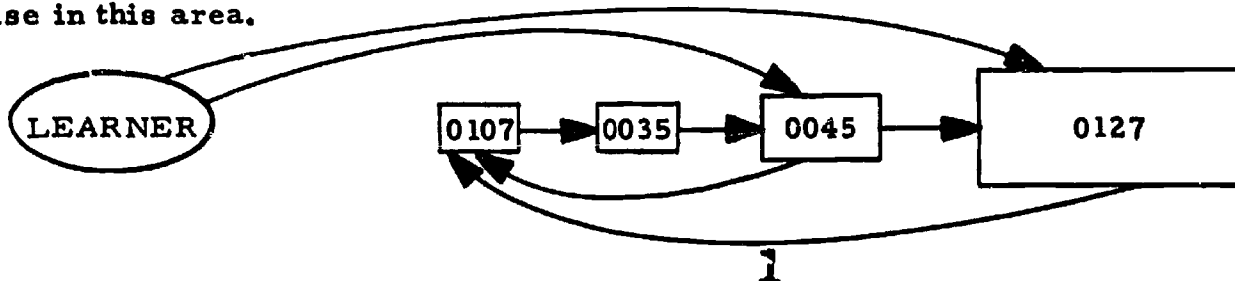


For example, the major challenging goal might be to obtain a radio amateur's license (an attractive and non-trivial goal for many persons). Once this objective has been set, it then becomes clear that the best approach to the major goal is to back off and master certain prerequisite goals, (learning to send and receive Morse code, studying prescribed theory, passing a specific written exam, etc.).

It is our feeling at Project Solo that such things as computer tutorials (or other teacher-authored activities) will stand or fall on the basis of their relevance to such a motivational strategy. It continues to be our estimate that the future lies with learner-controlled computing activity, especially if it is associated with major goals, but also if it is supported by thoughtful curricular structure (i.e. supported by well developed packages for achieving pre-requisite goals, including computer based tutorials, drills etc.).

Modules Related to Matrices

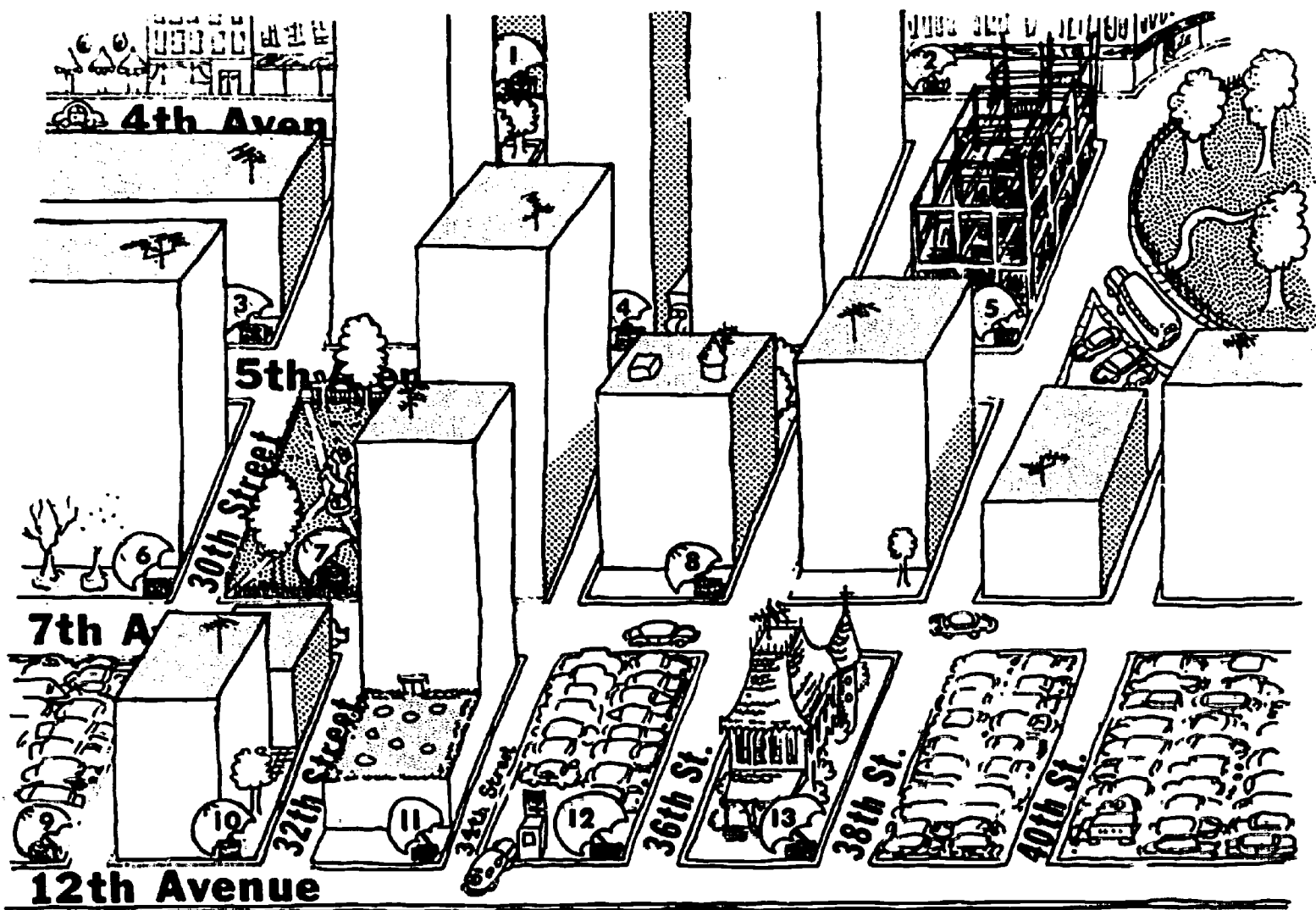
Newsletter No. 17 included a copy of module #0045 (Communication Matrices), and showed several "pre-requisite" paths that might lead to this unit. We are enclosing two such pre-requisite modules (#0107, Elementary Matrix Operations, and #0035, Matrix Multiplication). We are also enclosing module #0127 (Finite-State Automata) which goes a step beyond Communication Matrices. Thus for many students the sequence #0107, #0035, #0045 and #0127 could prove to be a well motivated but structured path to expertise in this area.



Elementary Matrix Operations

PROJECT SOLO
Department of Computer Science
University of Pittsburgh
(15213)

Module #0107



● This module introduces the idea of a matrix, and the elementary matrix operations of addition (page 2) and subtraction (page 4), and multiplication by a scalar (page 6).

● Module # 0036 covers matrix multiplication. Module #0045 discusses several modern applications of matrix multiplication.

The map on the cover of this module shows the streets and avenues of a section of the town of Munchin. As can be seen, there are employees of Col. Mayer's finger-licking Hot Dog Vending Company stationed at strategic intersections in the town. Being the business man that he is, Col. Mayer is interested in keeping an account of the number of sales of each of these vendors. He represents these sales figures in an ordered rectangular array as follows:

SALES FOR MONTH OF APRIL

	30 th St.	32 nd St.	34 th St.	36 th St.	38 th St.
4 th Avenue	0	877	0	1,996	0
5 th Avenue	1,652	0	2,008	0	1,765
7 th Avenue	3,077	2,437	0	2,981	0
12 th Avenue	3,618	3,051	4,171	4,632	3,987

This rectangular array is called a matrix (pronounced MAY-TRICKS). Each row of this matrix represents locations along a certain avenue. Each column of the matrix represents locations along a certain street. The entries of the matrix, which are called elements, represent the number of hot dogs sold by each vendor during the month of April.

Notice that the elements of the matrix show not only the number of sales but also the location of the vendor. For instance the entry 877 is the number of hot dogs sold by the vendor located at 4th Avenue and 32nd Street.

QUICK QUIZ

1. What is the location of the vendor whose sales totaled 4,171 for the month of April?

ANSWER: _____ Avenue and _____ Street.

2. How many hot dogs were sold in April by the vendor at 12th Avenue and 36th Street?

ANSWER: _____ hot dogs.

The 0 entries mean no sales because either there is no vendor at that corner or the vendor sold 0 hot dogs.

In the above matrix there are 4 rows (one for each of the avenues) and 5 columns (one for each street). The size of a matrix is given by the number of rows and columns. In this case the matrix is said to have size 4 by 5. In general, the size of a matrix is given as the number of rows by the number of columns.

MATRIX ADDITION

Another month has passed and Colonel Mayer has collected the sales of hot dogs by his vendors for the month of May. Here are the numbers:

SALES FOR MONTH OF MAY

	30 th St.	32 nd St.	34 th St.	36 th St.	38 th St.
4 th Avenue	0	1,002	0	1,890	0
5 th Avenue	1,723	0	2,971	0	1,887
7 th Avenue	4,007	2,983	0	3,623	0
12 th Avenue	3,773	3,076	5,380	5,179	4,837

Suppose we now want a table showing the total number of hot dogs sold by each vendor for the 2 months of April and May? It is probably obvious to you that the way to get this table is to add together each vendor's sales for both months. Doing this for the whole chart is called the operation of matrix addition.

Before going ahead and doing this, we first will introduce a more compact notation for matrices,* used by mathematicians. We will discard everything except the elements of the matrix, enclose them in square brackets, and give them a name as follows (capital letters are usually used):

$$A = \begin{bmatrix} 0 & 877 & 0 & 1,996 & 0 \\ 1,652 & 0 & 2,008 & 0 & 1,765 \\ 3,077 & 2,437 & 0 & 2,981 & 0 \\ 3,618 & 3,051 & 4,171 & 4,632 & 3,987 \end{bmatrix}$$

This is the matrix A which describes the sales for April

$$M = \begin{bmatrix} 0 & 1,002 & 0 & 1,890 & 0 \\ 1,723 & 0 & 2,971 & 0 & 1,887 \\ 4,007 & 2,983 & 0 & 3,623 & 0 \\ 3,773 & 3,076 & 5,380 & 5,179 & 4,837 \end{bmatrix}$$

This is the matrix M which describes the sales for May

To answer the last question (what are total sales of each vendor for April and May) we simply add the matrices A and M. This means that each element in A is added to the corresponding element in M.

*Matrices is the plural of matrix; it is pronounced MAY-TRA-SEES.

$$\begin{bmatrix} 0 & 877 & 0 & 1,996 & 0 \\ 1,652 & 0 & 2,008 & 0 & 1,765 \\ 3,077 & 2,437 & 0 & 2,981 & 0 \\ 3,618 & 3,051 & 4,171 & 4,632 & 3,987 \end{bmatrix}$$

← The matrix A

$$+ \begin{bmatrix} 0 & 1,002 & 0 & 1,890 & 0 \\ 1,723 & 0 & 2,971 & 0 & 1,887 \\ 4,007 & 2,983 & 0 & 3,623 & 0 \\ 3,773 & 3,076 & 5,380 & 5,179 & 4,837 \end{bmatrix}$$

← The matrix M

$$\begin{array}{c} \text{The matrix} \\ \text{A+M} \end{array} \Rightarrow \begin{bmatrix} 0+0 & 877+1,002 & 0+0 & 1,996+1,890 & 0+0 \\ 1,652+1,723 & 0+0 & 2,008+2,971 & 0+0 & 1,765+1,887 \\ 3,077+4,007 & 2,437+2,983 & 0+0 & 2,981+3,623 & 0+0 \\ 3,618+3,773 & 3,051+3,076 & 4,171+5,380 & 4,632+5,179 & 3,987+4,837 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1,879 & 0 & 3,886 & 0 \\ 3,375 & 0 & 4,979 & 0 & 3,652 \\ \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} \\ \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} \end{bmatrix}$$

EXERCISE

Fill in the missing entries in the sum matrix shown above. (Hard work? Cheer-up. You'll learn to use the computer for this purpose soon.)

MATRIX SUBTRACTION

Colonel Mayer wants to know the difference in hot dog sales between May and April for each vendor so that he can tell which location is showing the greatest boom (increase) in business.

The answer to this problem involves the matrix operation of subtraction. Matrix subtraction is similar to matrix addition, except that each element of A will be subtracted from the corresponding element of M. Thus,

$$M-A = \begin{bmatrix} 0 & 1,002 & 0 & 1,890 & 0 \\ 1,723 & 0 & 2,971 & 0 & 1,887 \\ 4,007 & 2,983 & 0 & 3,623 & 0 \\ 3,773 & 3,076 & 5,380 & 5,179 & 4,837 \end{bmatrix}$$

← The matrix M

$$- \begin{bmatrix} 0 & 877 & 0 & 1,996 & 0 \\ 1,652 & 0 & 2,008 & 0 & 1,765 \\ 3,077 & 2,437 & 0 & 2,981 & 0 \\ 3,618 & 2,051 & 4,171 & 4,632 & 3,087 \end{bmatrix}$$

← The matrix A

$$M-A = \begin{bmatrix} 0-0 & 1,002-877 & 0-0 & 1,890-1,996 & 0-0 \\ 1,723-1,652 & 0-0 & 2,971-2,008 & 0-0 & 1,887-1,765 \\ 4,007-3,077 & 2,983-2,437 & 0-0 & 3,623-2,981 & 0-0 \\ 3,773-3,618 & 3,076-2,051 & 5,380-4,171 & 5,179-4,632 & 4,837-3,087 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 125 & 0 & -106 & 0 \\ 70 & 0 & 963 & 0 & 122 \\ 930 & \underline{\hspace{1cm}} & 0 & \underline{\hspace{1cm}} & 0 \\ 155 & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & \underline{\hspace{1cm}} & 850 \end{bmatrix}$$

EXERCISE

1. Complete the above "difference" matrix.

In performing the matrix operations of addition and subtraction, notice that the 2 matrices being added or subtracted must be of the same size.

MATRIX MULTIPLICATION BY A SCALAR

With the approach of the summer months and the baseball season, Col. Mayer is predicting a 20% increase over the May sales for the month of June. What number of hot dogs does he predict each vendor to sell during the month of June?

The answer to this question involves the matrix operation of multiplication by a scalar. This means that each element in a matrix will be multiplied by the same number which is called a scalar. The scalar multiplier for our example is 1.20. We can say that each element in June is predicted to be 1.2 times as big as the corresponding element in May. The compact way of saying this with matrices is

$$J = 1.2 * M.$$

Which
means:

$$J = \begin{bmatrix} 1.20 * 0 & 1.20 * 1,002 & 1.20 * 0 & 1.20 * 1,890 & 1.20 * 0 \\ 1.20 * 1,723 & 1.20 * 0 & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} \\ 1.20 * 4,007 & \underline{\hspace{2cm}} & 1.20 * 0 & \underline{\hspace{2cm}} & 1.20 * 0 \\ 1.20 * 2,773 & 1.20 * 3,076 & 1.20 * 5,380 & 1.20 * 5,179 & 1.20 * 4,837 \end{bmatrix}$$

$$J = \begin{bmatrix} 0 & \underline{\hspace{2cm}} & 0 & \underline{\hspace{2cm}} & 0 \\ \underline{\hspace{2cm}} & 0 & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} \\ \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & 0 & \underline{\hspace{2cm}} & 0 \\ \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} & \underline{\hspace{2cm}} \end{bmatrix}$$

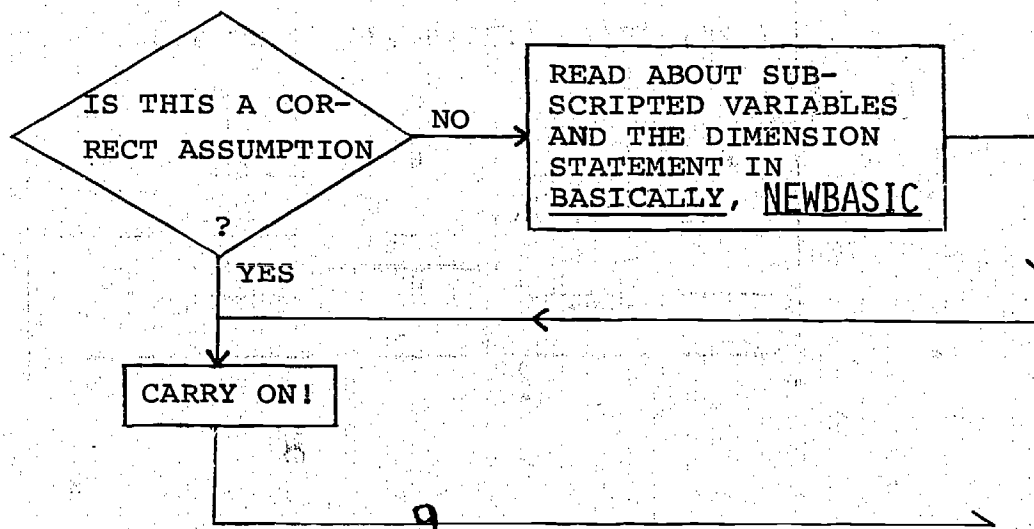
 EXERCISE (What, again!)

Complete the matrix $J=1.20*M$ shown on the previous page.
 (If you get tired, read on and find out how to use the computer to do this.)

Using the Computer to Add and Subtract Matrices and to Multiply Matrices by a Scalar

""""""""""
 "IMPORTANT NOTE:" Many versions of BASIC have special
 "matrix" commands that simplify working with matrices.
 However in this module we will not use these commands.
 There are two reasons for this. First, professional
 programmers do not use them, nor are they available in
 computer languages other than BASIC. Second, you will
 obtain much more insight into the mathematics of matrices
 by writing programs that manipulate them in the same
 way that the definitions say you should.

This module assumes that you understand what a subscripted variable is, and how such variables are used in BASIC.



STORING A MATRIX IN THE COMPUTER


Since an element of a matrix is located by its row number and column number (see page 2), the best way to store matrix elements is to use variables (locations) which have two subscripts. Then we can store the element which is in row i and column j in the location called $A(I,J)$. For example, $A(1,2)$ will be the variable name we give to the location where we store the matrix element in the first row, second column of A .

INPUT OF MATRIX ELEMENTS

The program segment shown below allows you to input the elements of a 2 by 3 (2×3) matrix. It expects you to input the numbers left to right, row by row.

```
105 DIM A(2,3)
110 FOR I=1 TO 2
115 FOR J=1 TO 3
120 INPUT A(I,J)
125 NEXT J
130 NEXT I
1000 END
RUN
?12
?-2
?14
?0
?33
?19
```

After you run this program, the memory of the computer looks like the following:



$A(1,1)$ 12	$A(1,2)$ -2	$A(1,3)$ 14
$A(2,1)$ 0	$A(2,2)$ 33	$A(2,3)$ 19

PRINTING OUT ELEMENTS

To print out the above matrix elements row by row we can add the following statements to the above program:

```
910 FOR I=1 TO 2
915 FOR J=1 TO 3
920 PRINT A(I,J);
925 NEXT J
930 PRINT (cr)
935 NEXT I
```

The semi-colon forces all the elements controlled by the inner FOR Loop to be printed on one line.

The extra PRINT statement forces the computer to print the next row on a new line.

SAMPLE PROGRAM - Enter and run this program for the Hot Dog Program (on p. 4).

Matrix Addition: $C=A+B$

```

1 DIM A(4,5), B(4,5), C(4,5)
5 PR. "THIS PROGRAM ADDS TWO 4 by 5 MATRICES"
10 PR. "INPUT THE ELEMENTS OF MATRIX A"
15 FOR I=1 TO 4
20 FOR J=1 TO 5
25 INPUT A(I,J)
30 NEXT J
35 NEXT I
40 PR. "NOW INPUT THE ELEMENTS OF MATRIX B"
45 FOR I=1 TO 4
50 FOR J=1 TO 5
55 INPUT B(I,J)
60 NEXT J
65 NEXT I
70 FOR I=1 TO 4
75 FOR J=1 TO 5
80 C(I,J)= A(I,J)+B(I,J)
85 NEXT J
90 NEXT I
95 PR. "THE SUM MATRIX IS:"
100 FOR I=1 TO 4
105 FOR J=1 TO 5
110 PRINT C(I,J);
115 NEXT J
120 PRINT
125 NEXT I
130 END

```

Input Matrix A

Input Matrix B

This segment of the program adds each element of B to the corresponding element of A, and puts the sum in the corresponding position of Matrix C

This prints out the Matrix C which contains the sum elements

????????????????????
CHALLENGE PROGRAM #1
 ?????????????????????

Write a program to calculate $C=A+B$ for matrices that can have any size up to and including 10 rows and 10 columns.

HINTS:

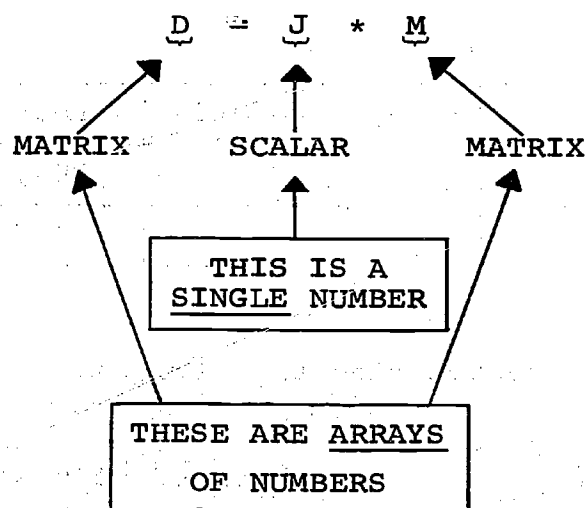
- Dimension A, B, and C as 10 by 10
- Allow the user to input the number of ROWS = M
- Allow the user to input the number of COLUMNS = N
- Change all the FOR loops to go TO M or TO N

????????????????????
 CHALLENGE PROGRAM #2
 ?????????????????????

Test the above program on your own 10 × 10 matrices.

????????????????????
 CHALLENGE PROGRAM #3
 ?????????????????????

Write a program which inputs a Matrix A and inputs a scalar factor J. The program should output the Matrix D which is the scalar product of J times M, i.e.



Test this program on the Hot Dog Problem on page 6.

????????????????????
 CHALLENGE PROGRAM #4
 ?????????????????????

(Confidential Note: We will throw in some technical language in this problem which will make it appear complicated. But the mathematics (and computing) is no harder than that used in the Hot Dog problem.)

The chart below shows the observed frequency values found in a series of pathologically verified cases of ulcer and cancer of the stomach:*

	Achlorhydria	Hypochlorhydria	Normal	Hyperchlorhydria
F= Chronic Ulcer	3	7	35	9
Stomach Cancer	22	2	6	0

The expected frequency values are given in the table E below:

	Achlorhydria	Hypochlorhydria	Normal	Hyperchlorhydria
E= Chronic Ulcer	16	5.8	26.4	5.8
Stomach Cancer	9	3.2	14.6	3.2

These tables can also be represented as matrices:

$$F = \begin{bmatrix} 3 & 7 & 35 & 9 \\ 22 & 2 & 6 & 0 \end{bmatrix} \quad (\text{observed values})$$

$$E = \begin{bmatrix} 16 & 5.8 & 26.4 & 5.8 \\ 9 & 3.2 & 14.6 & 3.2 \end{bmatrix} \quad (\text{expected values})$$

For statistical purposes we would like a table showing the differences between the observed and expected values. This means that you should modify your program to output the matrix $F-E$. HINT: As you might guess, all you have to do is modify the matrix addition program on page 9 so that line 80 uses a 'minus sign' (-). (Also change line 95.)

* See the book "Facts from Figures" by M. J. Horoney, pp. 246-270.

Partial Output for Program #4

$$F-E = \begin{bmatrix} -13.0 & 1.2 & ____ & ____ \\ ____ & ____ & ____ & ____ \end{bmatrix}$$

????????????????????
 CHALLENGE PROGRAM #5
 ????????????????????

Modify the above program so that a matrix of relative differences between observed and expected values is printed out, followed by a matrix of percentage differences.

Example for one element: (let's take the first row, second column)

$F(1,2) = 7$ (observed)

$E(1,2) = 5.8$ (expected)

The relative difference by which the observed value differs from the expected value for these particular elements is calculated by dividing the difference between F and E by E:

$$(7-5.8)/5.8 = 0.20689 = \text{RELATIVE DIFFERENCE}$$

The percentage difference is obtained by multiplying the relative difference by 100:

$$0.20689 \times 100 = 20.689\% = \text{PERCENTAGE DIFFERENCE}$$

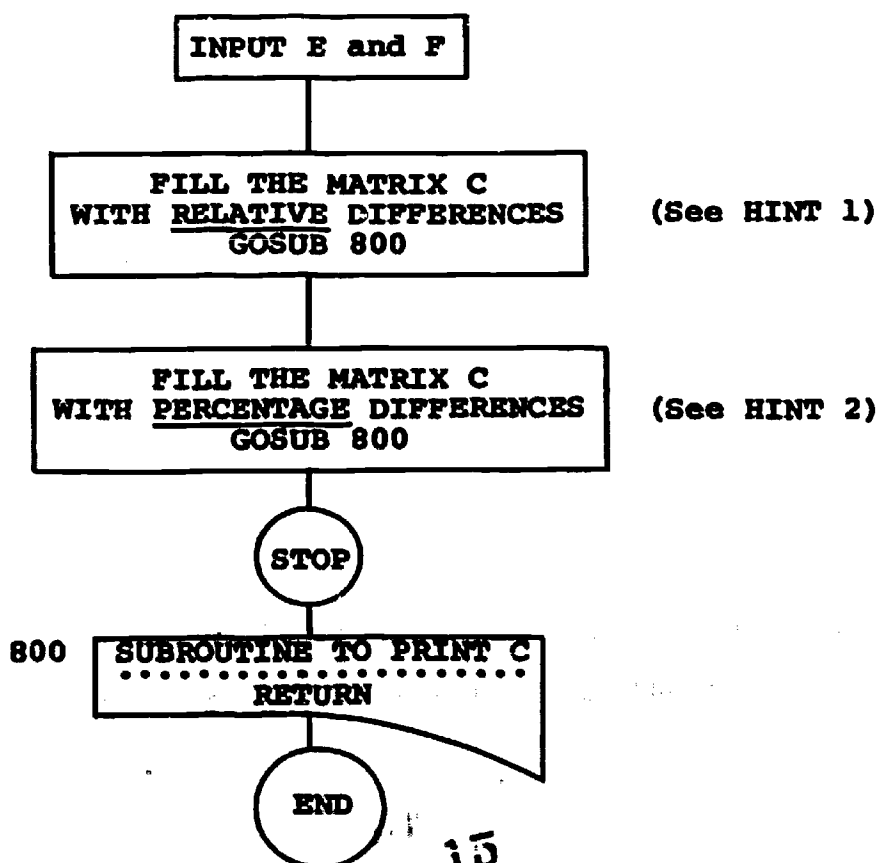
Now go ahead and do this for all elements in the matrices, using the computer as suggested on the next page.

HINT 1: Modify line 80 (p. 9) so that the Matrix C will contain relative differences.

$$C(I,J) = (F(I,J) - E(I,J)) / E(I,J)$$

HINT 2: For percentage differences do a scalar multiplication by 100. You can use the Matrix C "over again" (after printing the relative difference) by letting $C(I,J) = 100 * C(I,J)$ within a FOR loop.

HINT:3: Since you will want to print out two matrices (relative differences and percentage differences), but since they both will be called C (see HINT 2 above), it would be a good idea to put the print statements in a sub-routine as follows:



Matrix Multiplication

PROBLEM

$$\begin{matrix} & \text{A} & & & \text{B} & & & \text{C} \\ \begin{bmatrix} 5 & 4 & 2 \\ 3 & 9 & 8 \\ 6 & 2 & 1 \end{bmatrix} & * & \begin{bmatrix} 2 & 7 \\ 5 & 3 \\ 6 & 4 \end{bmatrix} & = & \begin{bmatrix} ? & ? \\ ? & ? \\ ? & ? \end{bmatrix} \end{matrix}$$



* means Matrix Multiplication!!?

SOLUTION

Step 1.

$$\begin{matrix} & \text{A} & & & \text{B} \\ \begin{bmatrix} 5 & 4 & 2 \\ 3 & 9 & 8 \\ 6 & 2 & 1 \end{bmatrix} & * & \begin{bmatrix} 2 & 7 \\ 5 & 3 \\ 6 & 4 \end{bmatrix} & = & \begin{matrix} \text{C} \\ \begin{bmatrix} 42 & ? \\ 99 & ? \\ 28 & ? \end{bmatrix} \end{matrix} \end{matrix}$$

Diagram illustrating the first row of matrix A multiplied by matrix B to produce the first row of matrix C:

- Row 1 of A: $5 \cdot 2 + 4 \cdot 5 + 2 \cdot 6 = 42$
- Row 2 of A: $3 \cdot 2 + 9 \cdot 5 + 8 \cdot 6 = 99$
- Row 3 of A: $6 \cdot 2 + 2 \cdot 5 + 1 \cdot 6 = 28$

That means I have to use real number multiplication (.) and addition (+) in just the right way.



Step 2.

$$\begin{matrix} & \text{A} & & & \text{B} \\ \begin{bmatrix} 5 & 4 & 2 \\ 3 & 9 & 8 \\ 6 & 2 & 1 \end{bmatrix} & * & \begin{bmatrix} 2 & 7 \\ 5 & 3 \\ 6 & 4 \end{bmatrix} & = & \begin{matrix} \text{C} \\ \begin{bmatrix} 42 & 55 \\ 99 & 80 \\ 28 & 52 \end{bmatrix} \end{matrix} \end{matrix}$$

Diagram illustrating the second column of matrix B multiplied by matrix A to produce the second column of matrix C:

- Column 1 of B: $5 \cdot 2 + 4 \cdot 5 + 2 \cdot 6 = 42$
- Column 2 of B: $5 \cdot 7 + 4 \cdot 3 + 2 \cdot 4 = 55$
- Column 3 of B: $3 \cdot 2 + 9 \cdot 5 + 8 \cdot 6 = 99$
- Column 4 of B: $3 \cdot 7 + 9 \cdot 3 + 8 \cdot 4 = 80$
- Column 5 of B: $6 \cdot 2 + 2 \cdot 5 + 1 \cdot 6 = 28$
- Column 6 of B: $6 \cdot 7 + 2 \cdot 3 + 1 \cdot 4 = 52$

I THINK I could use a computer...



MATRIX MULTIPLICATION

INTRODUCTION

The need to develop a compact notation for manipulating systems of linear equations led 19th century mathematicians to the discovery and use of matrices. (*Deep question: Are mathematical concepts discovered or invented?*) For the definition of a matrix and a discussion of matrix addition and multiplication of matrices by scalars see the module "ELEMENTARY MATRIX OPERATIONS". The present module is a presentation of another important matrix operation--matrix multiplication.

Multiplication of matrices was first suggested by research in the theory of linear equations, but it has turned out to be an important concept for many other applications, including multivariate statistics (*used by all social scientists*), analysis of communications within a group (*see module 0045 "COMMUNICATION MATRICES"*), electrical network problems, and mathematical models in psychology and economics.

Let's start with a SPECIAL CASE: *Multiplying A Matrix by a Vector.*

Consider the following set of three linear equations in three unknowns:

$$\begin{array}{rclcl} 4x_1 & + & 7x_2 & + & 12x_3 & = & 95 \\ 3x_1 & + & 2x_2 & + & 10x_3 & = & 58 \\ 11x_1 & + & 8x_2 & + & 5x_3 & = & 121 \end{array}$$

Matrix multiplication is defined in such a way that the above set of equations can be expressed in compact matrix form as follows:

$$A * X = B$$

Where: 1. "*" is the symbol for matrix multiplication
2. A is the matrix of coefficients

$$A = \begin{bmatrix} 4 & 7 & 12 \\ 3 & 2 & 10 \\ 11 & 8 & 5 \end{bmatrix}$$

3. B is the column vector of constant terms.
A column vector is a matrix with only one column.

$$B = \begin{bmatrix} 95 \\ 58 \\ 121 \end{bmatrix}$$

4. X is the column vector of unknowns.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

NOTE: Since X is a matrix its elements should have two subscripts--the first for the row and the second for the column--that is, strictly speaking we should write:

$$X = \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \end{bmatrix}$$

But since there is only one column there should be no confusion if we drop the second (column) subscript.

Notice that the product of a 3×3 matrix and a 3×1 matrix (column vector) is a 3×1 matrix (column vector) which is formed as follows:

Multiply each number in the first row of A by the corresponding number (first by first, second by second, third by third) in the first column of X (in this case X only has one column) and add these products. This gives the number in the first row and first column of $A * X$.

$$\begin{array}{c} \text{row 1} \end{array} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} * \begin{array}{c} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \\ \text{column 1} \end{array} = \begin{array}{c} \text{row 1 column 1} \\ \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ \cdot \\ \cdot \end{bmatrix} \end{array}$$

Do the same for the second row of A and the first, and only, column of X to determine the number in the second row and first column of $A * X$. That is:

$$\begin{array}{c} \text{row 2} \end{array} \begin{bmatrix} \cdot & \cdot & \cdot \\ a_{21} & a_{22} & a_{23} \\ \cdot & \cdot & \cdot \end{bmatrix} * \begin{array}{c} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \\ \text{column 1} \end{array} = \begin{array}{c} \text{row 2 column 1} \\ \begin{bmatrix} a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ \cdot \\ \cdot \end{bmatrix} \end{array}$$

Finally the number in the third row and the first column of $A * X$ is computed as follows:

$$\begin{array}{c} \text{row 3} \end{array} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{31} & a_{32} & a_{33} \end{bmatrix} * \begin{array}{c} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \\ \text{column 1} \end{array} = \begin{array}{c} \begin{bmatrix} \cdot \\ \cdot \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \end{bmatrix} \\ \text{row 3 column 1} \end{array}$$

To repeat the final result:

$$\begin{array}{c} \text{A} \\ \hline \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \end{array} \quad * \quad \begin{array}{c} \text{X} \\ \hline \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \end{array} = \begin{array}{c} \text{A*X} \\ \hline \begin{bmatrix} a_{11}x + a_{12}x + a_{13}x \\ a_{21}x + a_{22}x + a_{23}x \\ a_{31}x + a_{32}x + a_{33}x \end{bmatrix} \end{array}$$

NOTE: To keep track of what you are doing, it is sometimes helpful to point to the elements that are being multiplied.

For example, when you are computing the number in row 2 column 1 of $A * X$ above,

Your left hand points to:

$$\begin{bmatrix} a_{21} & a_{22} & a_{23} \end{bmatrix}$$

Your right hand points to:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

and you think of placing them side by side and combining them as follows:

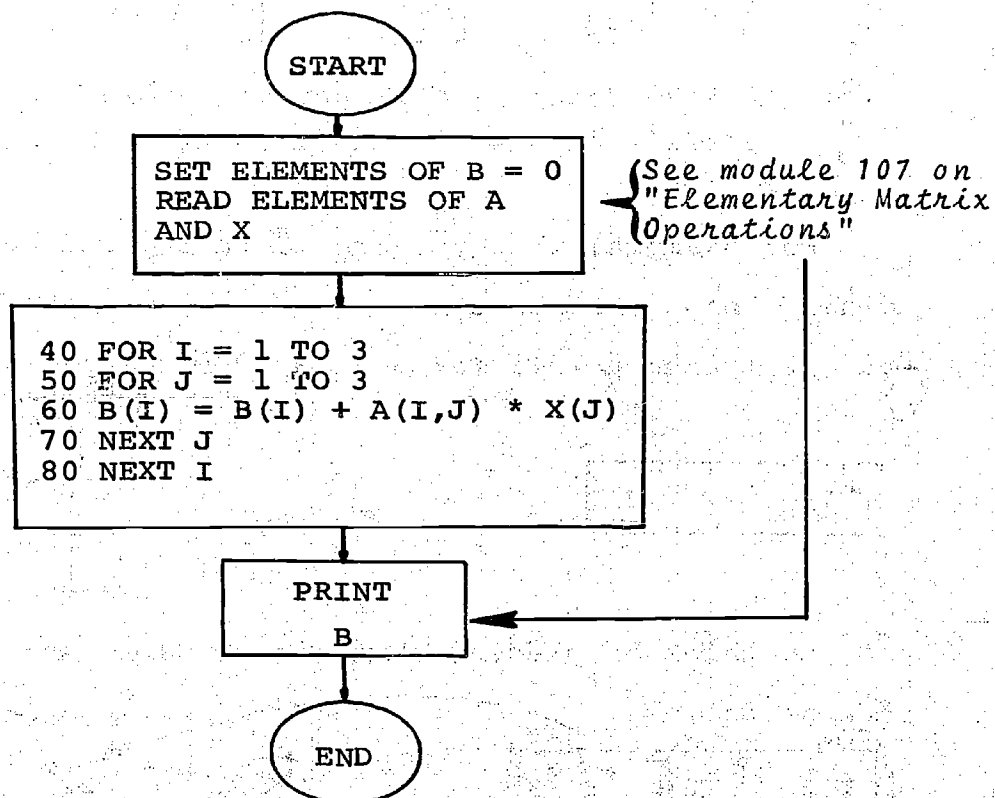
$$\begin{array}{|c|c|c|} \hline x_1 & x_2 & x_3 \\ \hline * & + & * \\ \hline a_{21} & a_{22} & a_{23} \\ \hline \end{array} \quad \rightarrow \quad \begin{bmatrix} a_{21}x + a_{22}x + a_{23}x \\ \text{etc.} \end{bmatrix}$$

Try it--you'll like it!

EXERCISE 1A

$$A * X = B$$

A program in BASIC which multiplies 3 x 3 matrix* by a 3 x 1 column vector might be organized as follows:



Write the complete program and try it with the following matrices:

$$N = 3, \begin{bmatrix} 3 & 1 & 7 \\ 9 & 5 & 2 \\ 2 & 4 & 8 \end{bmatrix} * \begin{bmatrix} -1 \\ 6 \\ 4 \end{bmatrix}$$

$$N = 3, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 20 \\ 20 \\ 20 \end{bmatrix}$$

Also multiply these by hand as well as by using your program.

* 3 x 3 is "pronounced" 3 by 3; the x does not mean multiplication.

EXERCISE 1B

In the above discussion there was nothing special about the number 3. The more general definition for multiplying a $N \times N$ matrix by a $N \times 1$ matrix (column vector) is:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n \end{bmatrix}$$

Write a BASIC program which reads a number N , a $N \times N$ matrix, and a $N \times 1$ matrix, computes the matrix product, and prints the result.

Programming Note: A feature of BASIC which is useful in dealing with matrices is SUBSCRIPTED VARIABLES. See your BASIC reference manual. You may also want to review FOR-NEXT loops. You will have to decide on a maximum possible value for N and write a DIM statement accordingly.

EXERCISE 1C

Test your program on the following data:

(a)

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix}$$

$N = 5$

(b)

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} * \begin{bmatrix} 21 & 73 & 84 & 96 \\ 18 & 98 & 41 & 32 \\ 26 & 47 & 93 & 41 \\ 22 & 27 & 68 & 74 \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$N = 4$

The General Case: More Than One Column

Suppose in the product $A * X = B$, A is a 3×3 matrix and X is a 3×2 matrix. Matrix multiplication is defined so that B will be a 3×2 matrix defined such that:

Each column of the product matrix B is the column vector that you get by multiplying A by the corresponding column of X .

This is illustrated very well on the cover of this module and it would be worth your time to study it carefully.

For those who like symbols, subscript, etc.

$$\begin{array}{c} 3 \text{ rows} \end{array} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{array}{c} * \end{array} \begin{array}{c} 2 \text{ columns} \end{array} \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix} =$$

$$\begin{array}{c} 3 \text{ rows} \end{array} \begin{bmatrix} a_{11}x_{11} + a_{12}x_{21} + a_{13}x_{31} & a_{11}x_{12} + a_{12}x_{22} + a_{13}x_{32} \\ a_{21}x_{11} + a_{22}x_{21} + a_{23}x_{31} & a_{21}x_{12} + a_{22}x_{22} + a_{23}x_{32} \\ a_{31}x_{11} + a_{32}x_{21} + a_{33}x_{31} & a_{31}x_{12} + a_{32}x_{22} + a_{33}x_{32} \end{bmatrix} \begin{array}{c} 2 \text{ columns} \end{array}$$

Matrix multiplication is defined for matrices A and X whenever the number of columns of A is equal to the number of rows of X . Another way to say this is that any $N \times K$ matrix can be multiplied by any $K \times M$ matrix. The product matrix will be $N \times M$ matrix.

The rule to remember is: $[N \times K] * [K \times M] = [N \times M]$

↑ ↑
must be the same

Matrices which can be multiplied are called conformable. Notice that it may be possible to multiply $A * X$ but not possible to multiply $X * A$. (Can you think of an example?)

EXERCISE 2A

Write a program which inputs three numbers N, K, M , a matrix A with N rows and K columns, and another matrix B with K rows and M columns. The program should output the matrix product $A * B$.

EXERCISE 2 B

Try your program on those products below which are conformable.

a) $N = 3, K = 3, M = 2$

$$\begin{bmatrix} 5 & 4 & 2 \\ 3 & 9 & 8 \\ 6 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 7 \\ 5 & 3 \\ 6 & 4 \end{bmatrix} = ?$$

b) $N = 3, K = 3, M = 3$

$$\begin{bmatrix} 2 & 0 & 2 \\ -5 & 3 & 6 \\ 7 & 1 & 8 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = ?$$

EXTRA: An element I (in a set with a multiplication operation defined on it) such that $I * A = A * I = A$ for any A is called a multiplicative identity. What is the multiplicative identity for $N \times N$ (square) matrices? What is the additive identity?

Answers for $N = 2$

MULTIPLICATIVE Identity = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

ADDITIVE Identity = $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

c) $N = 4, K = 4, M = 6$

$$\begin{bmatrix} 1 & 5 & 1 & 5 \\ -1 & 3 & -1 & 3 \\ 2 & 0 & 5 & 6 \\ -7 & 1 & 3 & 8 \end{bmatrix} * \begin{bmatrix} -3 & 1 & 0 & 2 & 4 & -2 \\ 6 & 4 & -5 & 12 & 8 & 3 \\ 7 & 6 & -1 & -3 & 5 & 9 \\ 0 & 1 & 5 & 9 & 7 & \end{bmatrix} = ?$$

The above two matrices are not conformable for multiplication in the reverse order. Hence, multiplication is not commutative for every pair of matrices. (That is, it is not true that $A * B = B * A$ for all matrices A and B .) But what if A and B are $N \times N$ (square) matrices? Do you think that matrix multiplication is commutative for square matrices?

d) $N = 2, K = 2, M = 2$

$$\begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} = ?$$

What can you conclude from (d) and (e)?

e) $N = 2, K = 2, M = 2$

$$\begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} = ?$$

f) $N = K = M = 3$

$$\begin{bmatrix} 1.000000 & 0.500000 & 0.333333 \\ 0.500000 & 0.333333 & 0.250000 \\ 0.333333 & 0.250000 & 0.200000 \end{bmatrix} * \begin{bmatrix} 9 & -36 & 30 \\ -36 & 192 & -180 \\ 30 & -180 & 180 \end{bmatrix}$$

LISTING OF A GENERAL PROGRAM TO MULTIPLY MATRICES

```

10 PRINT "THIS IS A PROGRAM TO MULTIPLY 2 MATRICES."
20 PRINT "THE MATRICES MUST BE CONFORMABLE & OF"
30 PRINT "SIZE 10 BY 10 OR LESS."
40 PRINT "INPUT THE DIMENSIONS OF THE MATRIX TO BE MULTIPLIED."
50 PRINT "WHAT IS THE NUMBER OF ROWS?"
60 INPUT R1
70 PRINT "WHAT IS THE NUMBER OF COLUMNS?"
80 INPUT C1
90 PRINT "INPUT THE DIMENSIONS OF THE MULTIPLIER MATRIX."
100 PRINT "WHAT IS THE NUMBER OF ROWS?"
110 INPUT R2
120 IF R2=C1 THEN 150
130 PRINT "THE MATRICES ARE NOT CONFORMABLE."
140 GOTO 90
150 PRINT "WHAT IS THE NUMBER OF COLUMNS?"
152 INPUT C2
155 PRINT "INPUT THE MATRIX TO BE MULTIPLIED."
160 PRINT "INPUT BY ROWS, ONE ELEMENT AT A TIME."
170 FOR I=1 TO R1
180 FOR J=1 TO C1
190 INPUT A(I,J)
200 NEXT J
210 NEXT I
220 PRINT "INPUT THE MATRIX THAT WILL BE THE MULTIPLIER."
230 PRINT "INPUT BY ROWS, ONE ELEMENT AT A TIME."
240 FOR I=1 TO R2
250 FOR J=1 TO C2
260 INPUT B(I,J)
270 NEXT J
280 NEXT I
300 FOR I=1 TO R1
310 FOR J=1 TO C2
320 LET C(I,J)=0
330 NEXT J
340 NEXT I
360 FOR K=1 TO C2
370 FOR I=1 TO R1
380 FOR J=1 TO R2
390 LET C(I,K)=C(I,K)+A(I,J)*B(J,K)
400 NEXT J
410 NEXT I
415 NEXT K
420 PRINT "THE PRODUCT MATRIX IS"
430 PRINT
440 FOR I=1 TO R1
450 FOR J=1 TO C2
460 PRINT C(I,J);
470 NEXT J
480 PRINT
490 NEXT I
500 PRINT
510 PRINT "DO YOU WISH TO DO ANOTHER MULTIPLICATION?"
520 PRINT "IF SO TYPE A 1 OTHERWISE TYPE A 2."
530 INPUT Y
540 IF Y=1 THEN 40
550 END

```

SAMPLE RUN OF /MAT/ USING THE DATA FROM THE COVER PICTURE

>RUN

THIS IS A PROGRAM TO MULTIPLY 2 MATRICES.

THE MATRICES MUST BE COMPATIBLE AND OF

SIZE 10 BY 10 OR LESS.

INPUT THE DIMENSIONS OF THE MATRIX TO BE MULTIPLIED.

WHAT IS THE NUMBER OF ROWS?

? 3

WHAT IS THE NUMBER OF COLUMNS?

? 3

INPUT THE DIMENSIONS OF THE MULTIPLIER MATRIX.

WHAT IS THE NUMBER OF ROWS?

? 3

WHAT IS THE NUMBER OF COLUMNS?

? 2

INPUT THE MATRIX TO BE MULTIPLIED.

INPUT BY ROWS, ONE ELEMENT AT A TIME.

? 5

? 4

? 2

? 3

? 9

? 8

? 6

? 2

? 1

INPUT THE MATRIX THAT WILL BE THE MULTIPLIER.

INPUT BY ROWS, ONE ELEMENT AT A TIME.

? 2

? 7

? 5

? 3

? 6

? 4

THE PRODUCT MATRIX IS

42 55

99 80

28 58

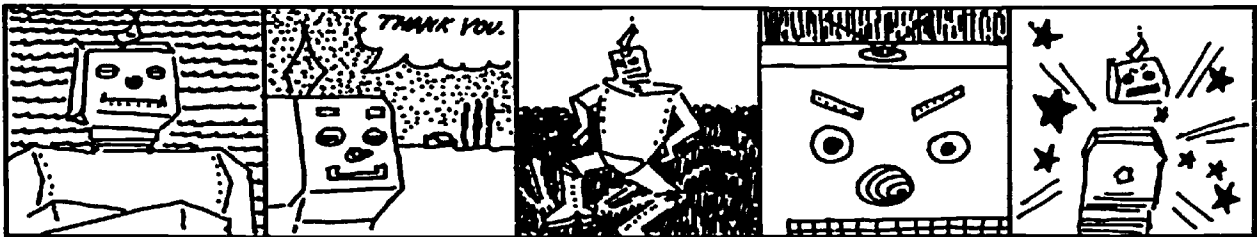
DO YOU WISH TO DO ANOTHER MULTIPLICATION?

IF SO TYPE A 1 OTHERWISE TYPE A 2.

? 2

>EXIT

Finite State Automata



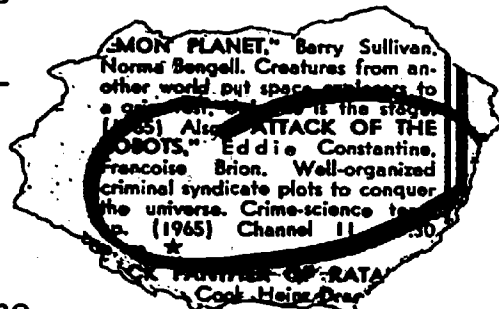
- This module will introduce you to a program that allows you to study, and if you wish, influence the "personality" of a robot.
- The model that the program uses to describe the robot is called a finite state automaton. Pages 6 to 9 show you how to describe such a machine (or automaton) with a diagram called a directed graph. (See also Module #0045.)
- You will be invited to use the finite state model to describe several other systems, and to write programs to simulate these systems.

**WANTED--EXPERIENCED ROBOT TO REPRESENT INTERNATIONAL ORGANIZATION.
MUST BE FLEXIBLE, AGGRESSIVE, ABLE TO MAKE DECISIONS. BOX 1759303.**

The idea of building robots has fascinated people for a long time.* An even more fascinating idea is to design machines that change in operation on the basis of "experience."

This module describes a simulated version of Rob the Robot, a computer program that behaves like such a machine. When you run this program, you have two choices. You can either 'live' with Rob the way he is, or you can help to re-shape his personality by giving him the benefit of your experience.

To use the program, login to the computer using the instructions given to you previously, then type: -NBS . In response to the NBS readiness symbol (>) type: RUN 106UOP /ROBOT-ROB/ .



* If you want to become a robot fan, I, Robot by Isaac Asimov is a science-fiction classic on the subject.

The program will then print the following:

IN THIS PROGRAM YOU CAN CHOOSE TO:

- 1) 'LIVE' WITH ROB AS HE IS;
- 2) CHANGE ROB'S PERSONALITY IMMEDIATELY;
- 3) CHANGE ROB'S PERSONALITY AFTER 3 MORE USERS CHOOSE THIS OPTION.

INPUT THE NUMBER OF YOUR CHOICE: 1,2,OR 3 ?

You should then type in the number of your choice.

Then a list of one-word descriptions of the robot's moods, treatments to be given him, and his reactions will be typed out:

ROB HAS SEVERAL MOODS:

- 1 FRIENDLY
- 2 INDIFFERENT
- 3 IRRITABLE
- 4 CO-OPERATIVE
- 5 HOSTILE

THE 'TREATMENTS' THAT CAN BE GIVEN TO ROB ARE:
SOMEONE.....HIM

- 1 HELPS
- 2 TEASES
- 3 COMPLIMENTS
- 4 ARM TWISTS
- 5 PUNCHES

ROB'S REACTION TO THE ABOVE 'TREATMENTS' CAN BE:
HE...

- 1 SMILES
- 2 STOMPS HIS FEET
- 3 SAYS 'THANK YOU'
- 4 KICKS YOU

There are 3 options within the program. In each of the cases you'll first be told the robot's present mood and that he was left in this mood by the last user of the program. You will then be asked to type in one of the treatments listed above.

Option 1

The computer will then respond by printing Rob's reaction to that treatment and his mood which probably changed as a result of the treatment given him. You'll be prompted to input another treatment to Rob in his new mood. This cycle of your inputting treatments and the computer outputting reactions and next mood continues for a total of ten inputs. Rob's final mood, that is the mood he'll be in for the next user is printed out and the program ends. The idea behind running option 1 is for you to study (but not change) Rob's personality.

SAMPLE

LET'S START...THE LAST USER LEFT ROB IN A FRIENDLY MOOD.
NOW TYPE IN A 'TREATMENT'... ?HELPS
WHEN SOMEONE HELPS FRIENDLY ROB, HE
SMILES AND BECOMES FRIENDLY

·
·
·

TREATMENT...?TEASES
IF SOMEONE TEASES ROB WHILE HE IS
CO-OPERATIVE HE STOMPS HIS FEET AND BECOMES INDIFFERENT

TREATMENT...?TEASES
OUR INDIFFERENT ROBOT STOMPS HIS FEET AND BECOMES
IRRITABLE WHEN SOMEONE TEASES HIM.

TREATMENT...?COMPLIMENTS
IF SOMEONE COMPLIMENTS ROB WHILE HE IS
IRRITABLE HE SAYS 'THANK YOU' AND BECOMES CO-OPERATIVE

YOU'VE LEFT ROB CO-OPERATIVE

LOGON AGAIN AT A LATER DATE TO SEE IF ROB HAS CHANGED ANY.
GOOD-BYE!

>

Option 2

After typing in a treatment you will be prompted to also input a reaction and a mood from the lists. These are what you expect the robot's responses will be. The computer will respond by printing 'agree' or 'disagree' followed by Rob's presently programmed reaction and next mood. Again you'll be prompted to input a treatment to the robot in his new mood and the cycle repeats 9 more times.

When you type in your expected responses you're suggesting reforms to Rob's personality which changes according to these suggestions. After the tenth cycle, you're given the chance to see the 'reformed' Rob in action. The program continues as if you had chosen option 1, that is you're prompted to just input a series of treatments. Here you have the chance to 'live' with Rob and his changed ways.

LET'S START...THE LAST USER LEFT ROB IN AN IRRITABLE MOOD.
NOW TYPE IN A 'TREATMENT'... ?HELPS
EXPECTED REACTION...?SAYS 'THANK YOU'
EXPECTED MOOD...?FRIENDLY
DISAGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
OUR IRRITABLE ROBOT SAYS 'THANK YOU' AND BECOMES
CO-OPERATIVE WHEN SOMEONE HELPS HIM.

TREATMENT...?TEASES
EXPECTED REACTION...?STOMPS HIS FEET
EXPECTED MOOD...?IRRITABLE
AGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
OUR CO-OPERATIVE ROBOT STOMPS HIS FEET AND BECOMES
IRRITABLE WHEN SOMEONE TEASES HIM.

TREATMENT...?HELPS
 EXPECTED REACTION...?SMILES
 EXPECTED MOOD...?CO-OPERATIVE
 DISAGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
 IF SOMEONE HELPS ROB WHILE HE IS
 IRRITABLE HE SAYS 'THANK YOU' AND BECOMES FRIENDLY

LET'S SEE WHAT THE 'REFORMED' ROB IS LIKE.
 ROB IS NOW FRIENDLY

TREATMENT...?COMPLIMENTS
 WHEN SOMEONE COMPLIMENTS FRIENDLY ROB, HE
 SMILES AND BECOMES FRIENDLY

.
 .
 .

Option 3

After typing in a treatment again, as in option 2, you will be prompted to input a reaction and a mood from the lists. Then you'll be prompted to input a treatment to the robot in his new mood and the cycle repeats. After ten inputs your expected responses will be recorded on a file. They will be saved, so that the robot's personality can be changed to conform to the influence of a group of suggestions, in other words to "social" pressure.

Finally, you're invited to logon again after the required number of users have chosen this option and to see what the poor mixed-up beast looks like now.

 SAMPLE

LET'S START...THE LAST USER LEFT ROB IN A CO-OPERATIVE MOOD.
 NOW TYPE IN A 'TREATMENT'... ?HELPS
 EXPECTED REACTION...?SMILES
 EXPECTED MOOD...?FRIENDLY
 DISAGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
 WHEN SOMEONE HELPS CO-OPERATIVE ROB, HE
 SAYS 'THANK YOU' AND BECOMES FRIENDLY

TREATMENT...?PUNCLES
THERE'S SOME MISUNDERSTANDING OR MISTYPING.

TREATMENT...?TEASES
EXPECTED REACTION...?STOMPS HIS FEET
EXPECTED MOOD...?IRRITABLE
AGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
IF SOMEONE TEASES ROB WHILE HE IS
CO-OPERATIVE HE STOMPS HIS FEET AND BECOMES IRRITABLE

TREATMENT...?COMPLIMENTS
EXPECTED REACTION...?SAYS 'THANK YOU'
EXPECTED MOOD...?CO-OPERATIVE
DISAGREE. ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT
WHEN SOMEONE COMPLIMENTS IRRITABLE ROB, HE
SMILES AND BECOMES FRIENDLY

PLEASE WAIT UNTIL THE COMPUTER PRINTS A 'GOOD-BYE'.
PLEASE WAIT

YOU'VE LEFT ROB FRIENDLY

LOGON AGAIN AT A LATER DATE TO SEE IF ROB HAS CHANGED ANY.
GOOD-BYE!

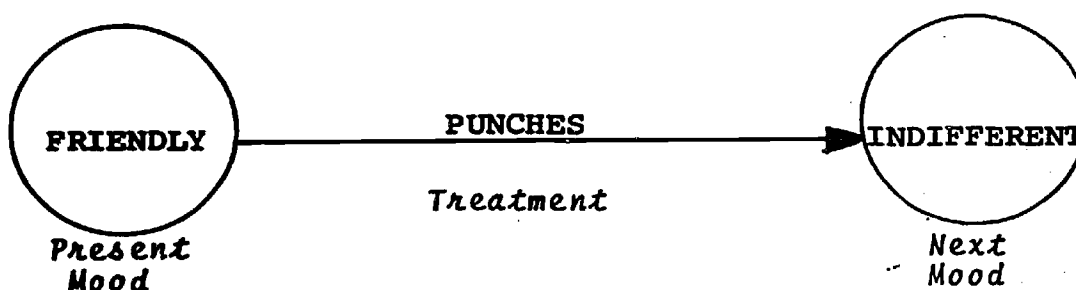
HOW ROB TICKS--THE FINITE STATE MACHINE AND THE STATE DIAGRAM

Rob's personality is described by what computer scientists call a finite-state machine. A sequential finite state machine consists of a finite number of states (moods), inputs (treatments), and outputs (reactions). The outputs (reactions) depend on what input (treatment) is given and on the present state (mood) of the machine. The next state (next mood) of the machine also depends on the input (treatment) it is given and on the present state (present mood).

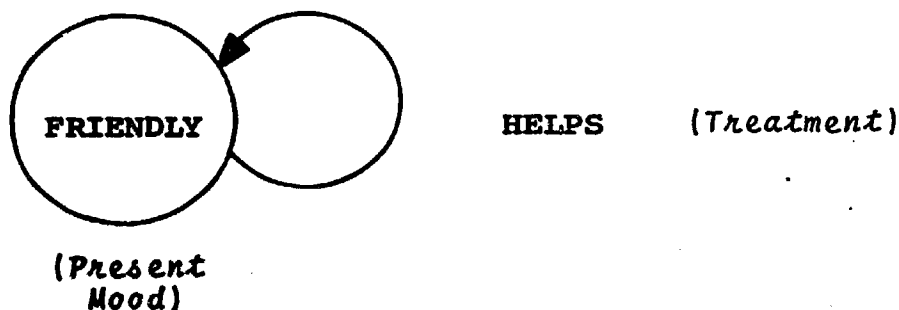
A finite state machine can be shown graphically by a state diagram. The state diagram consists of a directed

graph* with a few labels on each directed line. Let's look at Rob's one time personality by drawing its state diagram.

In Rob's state diagram each mood is represented by a circle in which is written the name of the mood. A directed line is drawn from one mood to another to represent a change from one mood to the other (the next mood). Since the next mood also depends on the treatment given in that mood, the treatment is written on the directed line.

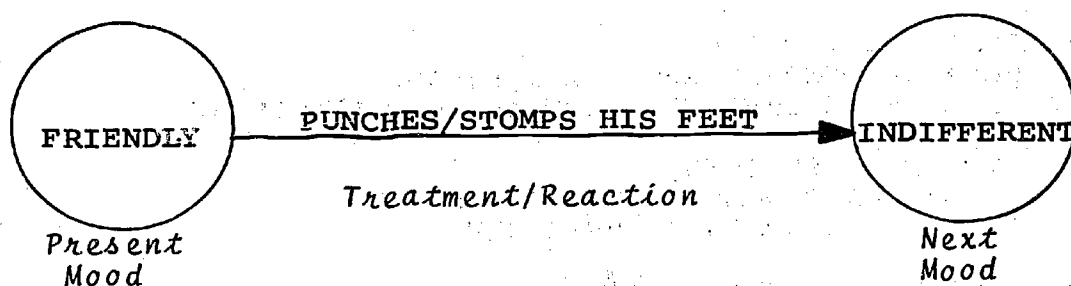


In some cases the mood does not change. This is represented by:

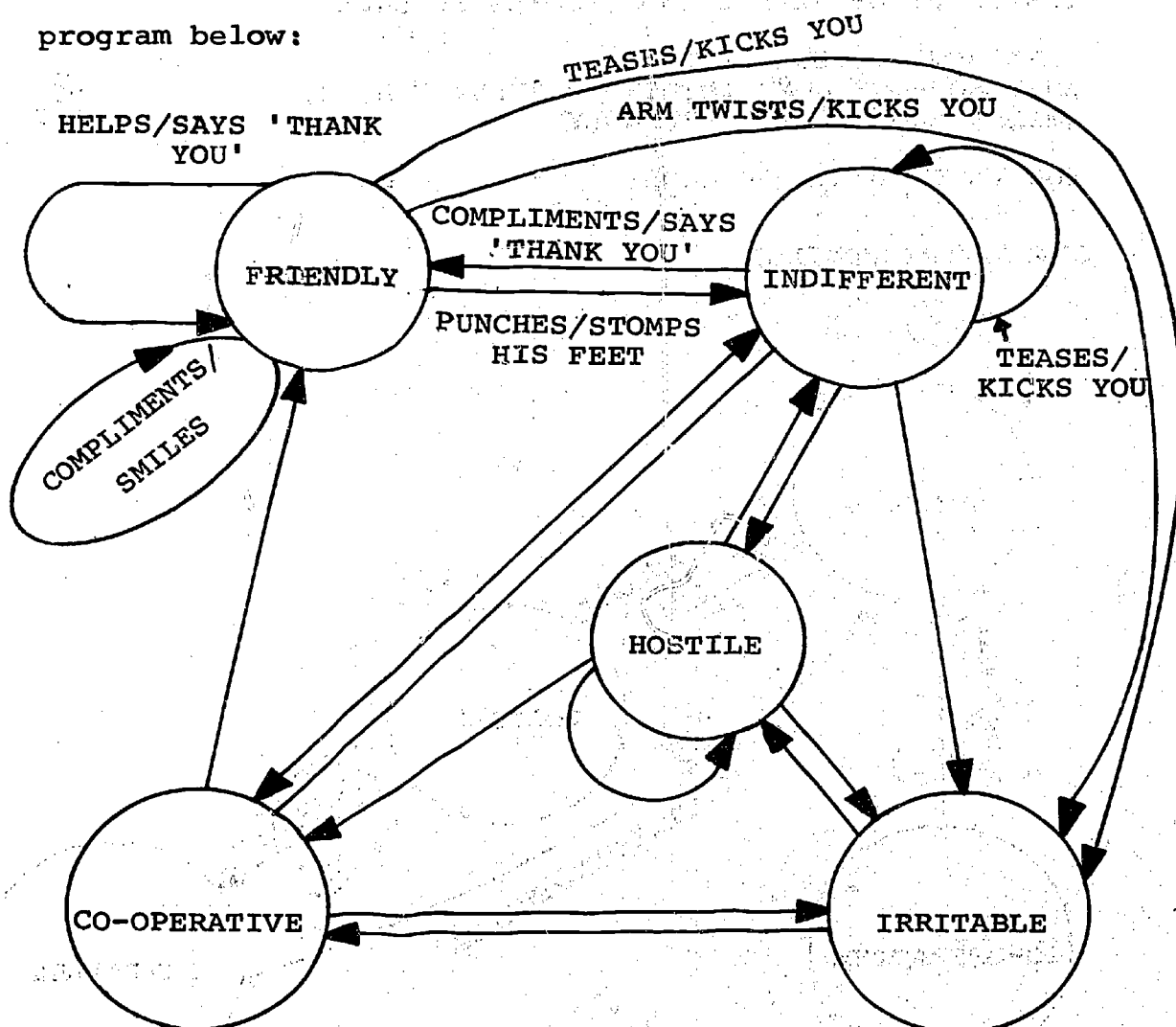


*See the module "Communications Matrices" for other examples of directed graphs. (Module #0045)

Since Rob's reaction depends on his present mood and treatment, his reaction is also printed on the directed line next to the corresponding treatment and present mood.



Rob's original personality is partly described by the program below:



Can you write in the missing treatment/reactions?

 SAMPLE COMPUTER EXCHANGE AND FLOW ON STATE-DIAGRAM

TREATMENT...?HELPS
 OUR CO-OPERATIVE ROBOT SAYS 'THANK YOU' AND BECOMES ①
 FRIENDLY WHEN SOMEONE HELPS HIM.

TREATMENT...?PUNCHES
 IF SOMEONE PUNCHES ROB WHILE HE IS ②
 FRIENDLY HE STOMPS HIS FEET AND BECOMES HOSTILE

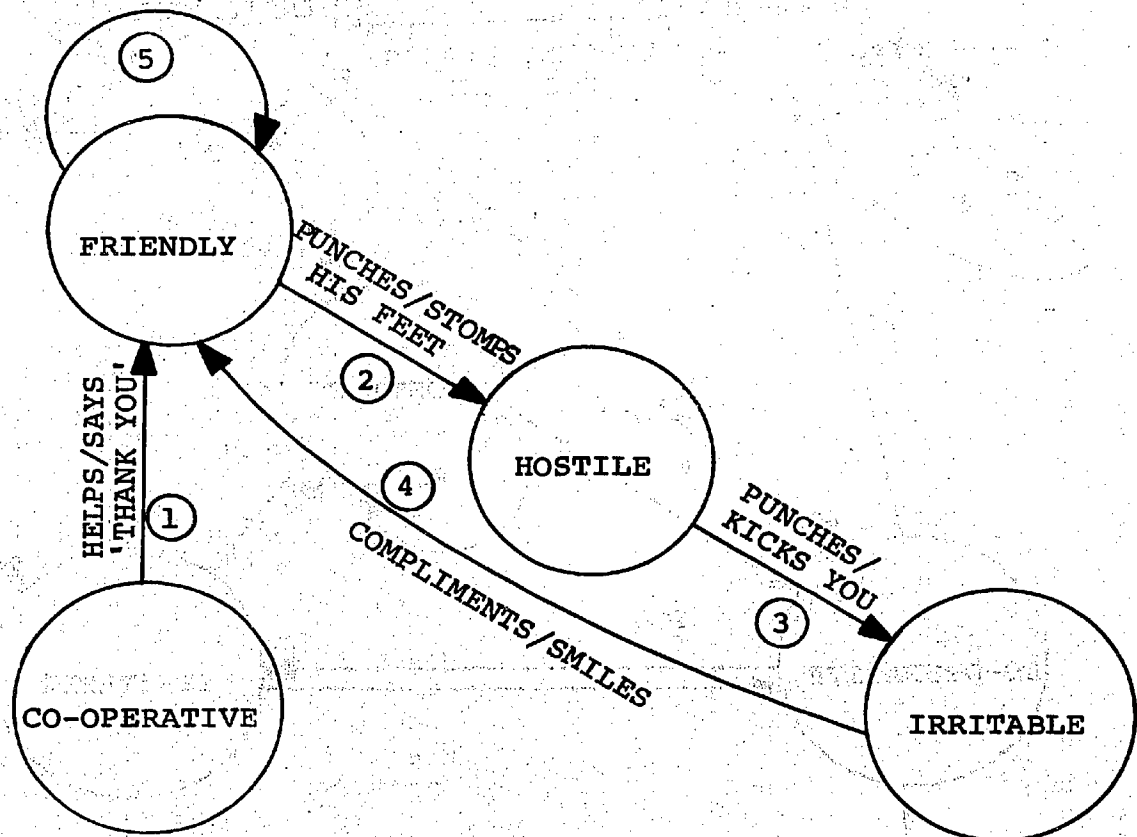
TREATMENT...?PUNCHES
 WHEN SOMEONE PUNCHES HOSTILE ROB, HE ③
 KICKS YOU AND BECOMES IRRITABLE

TREATMENT...?COMPLIMENTS
 IF SOMEONE COMPLIMENTS ROB WHILE HE IS ④
 IRRITABLE HE SMILES AND BECOMES FRIENDLY

TREATMENT...?COMPLIMENTS
 WHEN SOMEONE COMPLIMENTS FRIENDLY ROB, HE ⑤
 SMILES AND BECOMES FRIENDLY

YOU'VE LEFT ROB FRIENDLY

COMPLIMENTS/SMILES



MODEL BUILDING WITH THE STATE DIAGRAM

Now try your hand at designing models which can be represented by state diagrams.

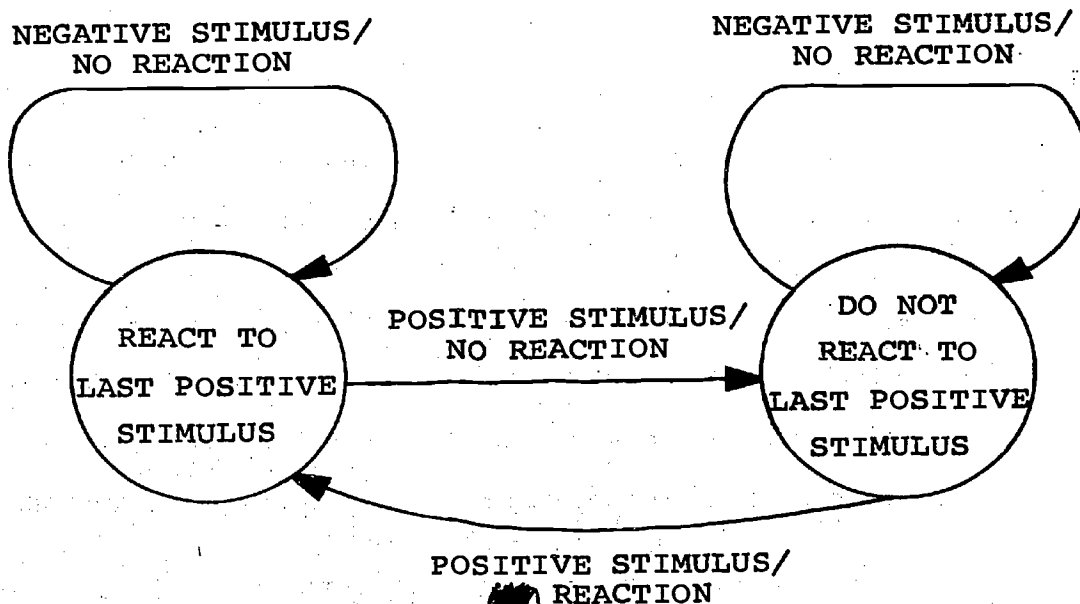
??????????
 ?PROBLEMS*?
 ??????????

1. An organism is excited by stimuli of 2 types-- "positive" and "negative"; the organism shows no reaction to negative stimuli and reacts to alternate positive stimuli.

The inputs are: positive stimulus, negative stimulus.
 The outputs are: reaction, no reaction.
 The states are: reaction to last positive stimulus,
 no reaction to last positive stimulus.

Draw the state diagram to illustrate this situation.

 *SAMPLE*SOLUTION



*The problems given here are taken from Introduction to the Theory of Finite State Machines by Arthur Gill. Because they were meant for "hand" solution, they are not too interesting. Readers who contemplate simulating these systems on computers (see problem 4), should first "improve" the problem

2. The direction of motion of a motor-driven wheel is controlled by a 2-position switch. The right position causes a clockwise rotation of the wheel; the left position causes a counterclockwise rotation. Each time the wheel changes direction, an indicator lamp flashes.

The inputs are: right position on switch, left position on switch.

The outputs are: Lamp on, Lamp off.

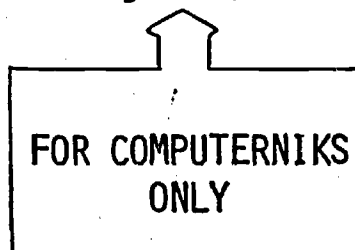
The states are: Clockwise rotation, counterclockwise rotation.

Draw the state diagram to illustrate this situation.

3. A freight elevator serving a 3-story warehouse with a call button on each floor operates according to the rules: If a single button is pushed, the elevator moves to the floor on which the pushed button is located; if 2 or 3 buttons are pushed simultaneously, the elevator moves to the lowest calling floor. No button can be pushed when the elevator is in motion.

This operation can be represented by the basic finite-state model. Enumerate the inputs, outputs and states. Draw the state diagram which represents the situation.

4. Study the listing and explanation of the program /ROBOT-ROB/ on the next few pages, and then write a program to simulate either the elevator in problem 3, or an elevator of your own design... (or some other system!)



4.3 PROGRAM DESCRIPTION (TECHNICAL ASPECTS)

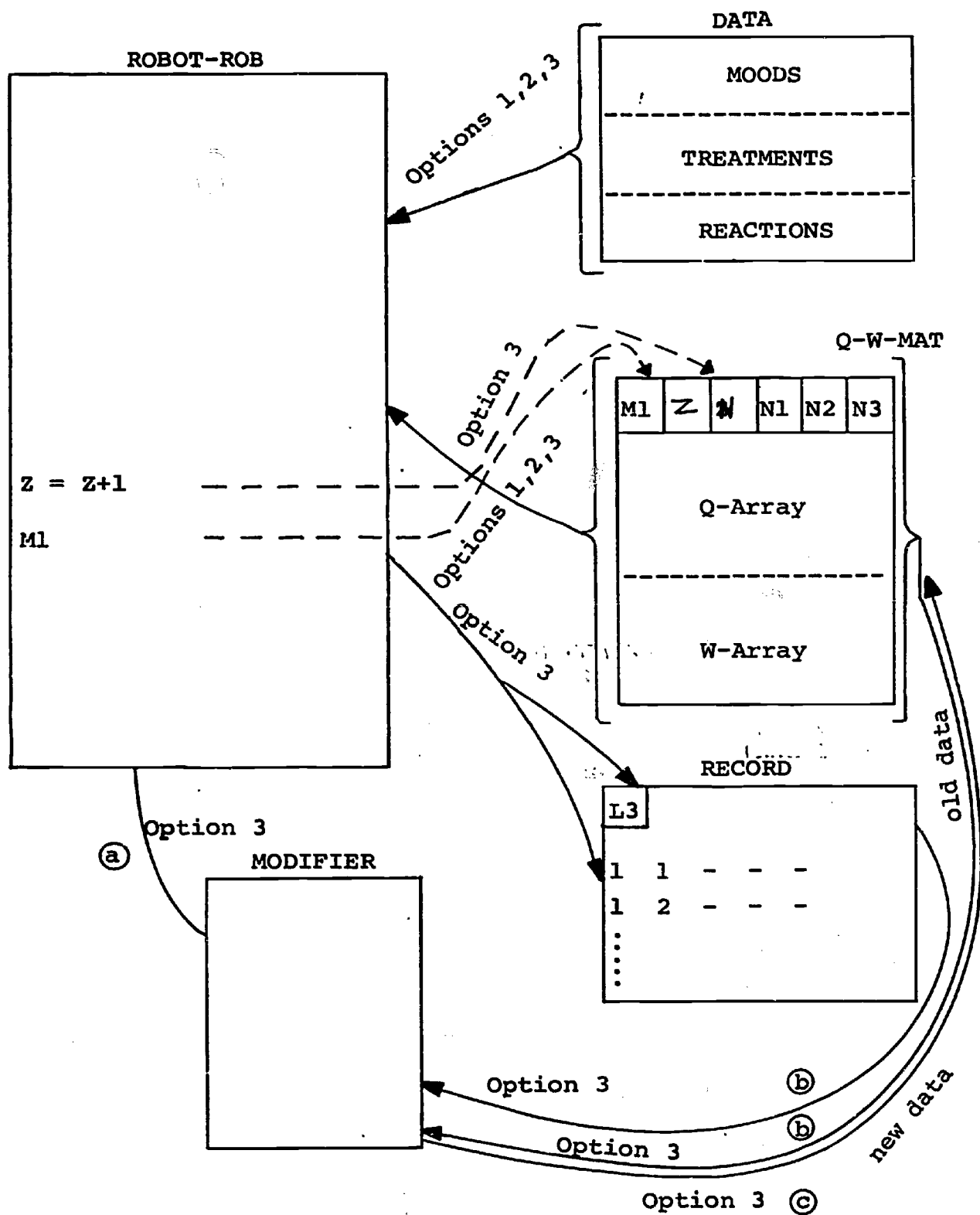
4.31 Exchange Between Files

The coding of the program was done in the NEWBASIC language of the Com-Share system. The main program ROBOT-ROB manages several data and program files. Figure 5 shows diagrammatically the exchange between program and data files. ROBOT-ROB is the controlling program until the MODIFIER program file is accessed (a in Figure 5). At that point the MODIFIER program controls the opening, reading from (b in Figure 5), writing onto (c in Figure 5) and the closing of the data files.

The main program initially reads from the DATA and Q-W-MAT data files. DATA is a symbolic file on which is stored the alphanumeric representation of the moods, treatments and reactions. Each set of information is read into a different string array. Q-W-MAT is a random access file. On it is kept the elements of the Q-array (the next mood ^{mapping} function*), the w-array (the reaction ^{mapping} function), the index number of the last mood in which the robot was left (M1), the number of users who have chosen option 3 (Z), and the number of users who must choose option 3 before the modification takes place (N). The random access file is used even though it is read sequentially because the system does not permit writing on a symbolic file without destroying

* See section 4.2.

EXCHANGE BETWEEN FILES



40 Figure 5

information previously written. The random access file allows writing and re-writing at specific locations without destroying the information at other locations.

The program files C-DATA and C-Q-W-MAT were used to initially write the two data files. A listing and sample execution of these programs and a listing of the data files created are shown below and on the following pages.

```
-COPY /C-DATA/ TO TEL
5 STRING A(10), B(10), C(10)
10 PR."TYPE IN N1";
15 INPUT N1
20 INPUT A(I) FOR I=1 TO N1
25 PR."TYPE IN N2";
30 INPUT N2
35 INPUT B(I) FOR I=1 TO N2
40 PR."TYPE IN N3";
45 INPUT N3
50 INPUT C(I) FOR I=1 TO N3
55 OPEN /DATA/ FOR OUTPUT AS 2
60 PRINT ON 2, N1
65 FOR I=1 TO N1
70 PRINT ON 2, A(I)
75 NEXT I
80 PRINT ON 2, N2
85 FOR I=1 TO N2
90 PRINT ON 2, B(I)
95 NEXT I
100 PRINT ON 2, N3
105 FOR I=1 TO N3
110 PRINT ON 2, C(I)
115 NEXT I
120 CLOSE 2
125 END
```

Creates /DATA/ file

>RUN /C-DATA/

TYPE IN N1 ?5
 ?FRIENDLY
 ?INDIFFERENT
 ?IRRITABLE
 ?CO-OPERATIVE
 ?HOSTILE
 TYPE IN N2 ?5
 ?HELPS
 ?TEASES
 ?COMPLIMENTS
 ?ARM TWISTS
 ?PUNCHES
 TYPE IN N3 ?4
 ?SMILES
 ?STOMPS HIS FEET
 ?SAYS 'THANK YOU'
 ?KICKS YOU

/DATA/ as used in the
 main program:

COPY /C-Q-W-MAT/ TØ TPT
 5 INTEGER Q(5,5), W(5,5)
 10 INTEGER Z, N1, N2, N3, ~~N~~ ^N
 15 PR. "TYPE IN Z, N1, N2, N3" ^{M1, N}
 20 INPUT Z, N1, N2, N3 ^{MAPPING}
 25 PR. "INPUT Q-MATRIX (NEXT STATE ~~PR~~)"
 30 FØR I=1 TØ N1
 35 INPUT Q(I,J) FOR J=1 TØ N2
 40 NEXT I
 45 PR. PR. "INPUT W-MATRIX (RESPØNSE ~~PR~~)" ^{MAPPING}
 50 FOR I=1 TØ N1
 55 INPUT W(I,J) FOR J=1 TØ N2
 60 NEXT I
 65 OPEN /Q-W-MAT/ RANDØM ØUTPUT 2
 70 L=Ø
 75 PRINT ØN 2 AT L: Z, N1, N2, N3 ^{M1, N}
 80 L=L+Ø
 85 FØR I=1 TØ N1
 90 PRINT ØN 2 AT L: Q(I,J), FOR J=1 TØ N2
 95 NEXT I
 100 FOR I=1 TØ N1
 105 PRINT ØN 2 AT L: W(I,J) FOR J= 1 TØ N2
 110 NEXT I
 115 CLOSE 2
 120 END

```

-CREATE /Q-W-MAT/
-NBS
VER. MAY 24 13:01
>RUN /C-Q-W-MAT/

TYPE IN m, u Z, N1, N2, N3
? 00554
0 5 5 4
INPUT Q-MATRIX (NEXT STATE mapping FUNCTION)
?1 3 1 3 2
?4 2 1 3 5
?4 5 4 5 5
?1 3 1 3 2
?4 5 2 5 3

INPUT W-MATRIX (RESPONSE mapping FUNCTION)
?3 4 1 4 2
?1 4 3 2 2
?3 2 1 2 2
?3 2 1 2 4
?1 2 1 2 4

```

The data file which is opened for output under option 3 is RECORD. It is a random access file on which is recorded the user's expected reactions and next moods for the robot. These expected responses are recorded along with the corresponding present mood-treatment combination which elicited the expected responses. The data is structured as a set of $N1 \times N2^*$ linked lists. Each element of the list has 5 fields:

```

present mood
treatment
expected reaction
expected next mood
link

```

* $N1$ is the total number of moods and $N2$ is the number of treatments. $N1 \times N2$ represents the number of all possible mood-treatment combinations.

When initially constructed by the program C-RECORD, only of each list element are specified. The remaining 3 fields the first 2 fields are given a value of 0. When initially written the RECORD file looks like:

1	1	0	0	0
1	2	0	0	0
1	3	0	0	0
	:			
1	N2	0	0	0
	:			
N1	N2	0	0	0

A "free space" location counter (L3) is set at location 0. Its value is $5 \cdot N1 \cdot N2 + 1$, the location of the first field after the last link field.

When a user has chosen option 3 and he encounters a certain mood-treatment combination, the appropriate list is located and the numbers corresponding to his expected responses are written in the reaction--next mood fields, if free. If these locations are not free because he or another user had encountered this mood-treatment combination previously, the list is sequentially searched for its last element. Then the location of the next free space is written into the link field of this element and the information is written beginning at the free location, adding another element to this list. Then the free space location counter is incremented by 5.

A listing of C-RECORD which creates the RECORD file is given below:

```

5 INTEGER I,J,K,L
6 INPUT N1,N2
10 K=0 LET L=1
15 OPEN /RECORD/ RANDOM OUTPUT 2
20 FOR I=1 TO N1
25 FOR J=1 TO N2
30 PRINT ON 2 AT L: I,J,K,K,K
35 NEXT J
36 NEXT I
40 K=N1*N2*5 +1
45 L=L+1
50 PRINT ON 2 AT L:K
55 CLOSE 2
60 END

```

Its only input (line 6) are N1, the number of moods and N2, the number of treatments.

The following program reads from the RECORD file and prints the values of each set of information. This program was composed just to read from the RECORD file for this illustration.

```

5 INTEGER A(625), L,L3
10 OPEN /RECORD/ RANDOM INPUT 8
15 L=0
20 INPUT FROM 8 AT L:L3
25 PR:"L3= ":L3
3
  0 L=1
35 FOR J=1 TO L3-1
40 INPUT FROM 8 AT L:A(J)
45 NEXT J
50 CLOSE 8
55 FOR I=1 TO (L3-1)/5
60 PRINT A(J); FOR J= 5*(I-1)+1 TO 5*(I-1)+5
65 PR.
70 NEXT I

```

The RECORD file after several people have exchanged with ROBOT-ROB and chose option 3 is shown on the next page.

L.3=	511					
1	1	1	1	156		
1	2	2	3	161		Location of next free space
1	3	1	1	201		
1	4	2	3	171		
1	5	2	5	131		
2	1	2	4	0		
2	2	2	3	136		
2	3	1	4	331		
2	4	2	3	231		
2	5	4	5	0		
3	1	1	2	191		
3	2	0	0	0		
3	3	3	2	166		
3	4	0	0	0		
3	5	4	5	126		
4	1	2	4	151		
4	2	4	3	371		
4	3	3	1	221		
4	4	2	5	291		
4	5	2	3	326		
5	1	1	1	146		
5	2	4	5	0		
5	3	3	2	381		
5	4	2	5	396		
5	5	2	5	241		
5		2	5	141		

(Break in listing)

...

4	1	3	1	0
1	3	3	1	506
1	4	2	3	0
2	5	2	5	0
5	4	2	5	0
5	1	3	4	0
4	3	2	4	0
1	3	1	1	0

4.32 The Modification Process

The last file that the main program controls is a program file, MODIFIER. After the expected responses of the user who had chosen option 3 are written onto RECORD, the program checks if the specified amount of data has been collected for the modification of the program's model. If so, the main program loads the MODIFIER program and initiates its execution.

It reads in the present values of the Q-array (next mood ^{mapping} ~~function~~) and W-array (reaction ^{mapping} ~~function~~) from Q-W-MAT. Then for each of the possible mood-treatment combinations (which is a total of $N1 \times N2$) it reads all of the expected responses (next state and reaction) that have been recorded for the combination.

If the number of times a particular reaction is recorded is one-half or more times the total number of times the mood-treatment combination has been encountered, the value of the corresponding W-array element is changed to the value of the recorded reaction. The same criteria is used for changing the elements of the Q-array, using the recorded (expected) next moods.

The new values of the Q and W arrays are written onto the Q-W-MAT file overwriting the previous values. The "number of users" counter (Z) is reset to 0 as is the "number of users required to effect a modification" counter (N). The RECORD file is erased and recreated as in C-RECORD, although this process is now a part of the MODIFIER program.

A listing of the MODIFIER program is given below:

```

COPY /MODIFIER/ TO TEL
1 NORMAL MODE IS INTEGER
5 INTEGER Q(5,5),W(5,5),U(10),T(10)
6 REAL C
10 OPEN /Q-W-MAT/ RANDOM INPUT AS 2
15 L=3
20 INPUT FROM 2 AT L:N1,N2,N3
30 FOR I=1 TO N1
35 INPUT FROM 2 AT L: Q(I,J) FOR J=1 TO N2
40 NEXT I
45 FOR I=1 TO N1
50 INPUT FROM 2 AT L: W(I,J) FOR J=1 TO N2
55 NEXT I
6
0 CLOSE 2
63 PR. PR."PLEASE WAIT UNTIL THE COMPUTER PRINTS 'GOOD-BYE'"
65 OPEN /RECORD/ RANDOM INPUT AS 3
70 L=0
75 INPUT FROM 3 AT L:L3
85 FOR I=1 TO N1
90 FOR J=1 TO N2
91 LET T(I1) = 0 FOR I1=1 TO N3 !TALLYS RESPONSES
9
3 LET U(I1) = 0 FOR I1=1 TO N1 !TALLYS NEXT STATE
95 LET C=0 !C COUNTS NO. OF TIMES STATE & INPUT WERE ENCOUNTERED
100 L=(I-1)*N2*5 + (J-1)*5 + 1
105 INPUT FROM 3 AT L: M,M1,R,S
110 IF R=0 GOTO 190
115 T(R) = T(R)+1
1
20 U(S) = U(S)+1
125 C=C+1
130 INPUT FROM 3 AT L: S2
135 IF S2=0 GOTO 155
140 L=S2+2
145 INPUT FROM 3 AT L:R,S
150 GOTO 115
155 C=.5*C
160 FOR I1=1 TO N3 !CHANGE RESPONSES
165 IF T(I1)>=C LET W(I,J) =I1, GOTO 175
170 NEXT I1
175 FOR I1=1 TO N1 !CHANGE NEXT STATE
180 IF U(I1)>=C LET Q(I,J)=I1, GOTO 190
185 NEXT I1
190 NEXT J
195 NEXT I
200 CLOSE 3
2
05 OPEN /Q-W-MAT/ RANDOM OUTPUT 2

```

```

210 L=1
212 Z=0 LET N=0
215 PRINT ON 2 AT L:Z,N
220 L=6
225 FOR I=1 TO N1
230 PRINT ON 2 AT L: Q(I,J) FOR J=1 TO N2
235 NEXT I
240 FOR I=1 TO N1
245 PRINT ON 2 AT L: W(I,J) FOR J=1 TO N2
250 NEXT I
255 CLOSE 2
260 K=0 LET L=1
265 OPEN /RECORD/ RANDOM OUTPUT 4
270 ERASE 4 FROM 0 TO L3
275 FOR I=1 TO N1
280 FOR J=1 TO N2
285 PRINT ON 4 AT L:I,J,K,K,K
290 NEXT J
295 NEXT I
300 K=N1*N2*5 + 1
305 L=0
310 PRINT ON 4 AT L:K
315 CLOSE 4
320 PR. PR."LOGON AGAIN TO SEE THE 'REFORMED' ROBOT-ROB."
325 PR."GOOD-BYE"
330 END

```

The following is a listing of the program ROBOT-ROB.

```

-COPY /ROBOT-ROB/ TO TEL
5 NORMAL MODE IS INTEGER
10 STRING A(5),B(5),C(5)
15 INTEGER S(10),R(10),U(10),V(10),Q(5,5),W(5,5)
20 OPEN /Q-W-MAT/ RANDOM INPUT 2
25 L=0
30 INPUT FROM 2 AT L: M1,Z,N
35 CLOSE 2
40 IF N>0 GOTO 75
45 PR." ROB'S PERSONALITY HAD JUST BEEN CHANGED. IT'S NOW"
50 PR."YOUR CHOICE TO DETERMINE HOW MANY USERS YOU WOULD LI
55 PR."TO SUGGEST MODIFICATIONS TO ROB'S PRESENT PERSONALIT
60 PR."I.E., HOW MANY PEOPLE WILL USE THE PROGRAM BEFORE RO
65 PR.""CHANGES' AGAIN?"
70 INPUT N
71 OPEN /Q-W-MAT/ RANDOM OUTPUT 2
72 L=2
73 PRINT ON 2 AT L: N
74 CLOSE 2
75 N4=N-Z-1
80 PR." IN THIS PROGRAM YOU CAN HELP TO MOLD THE PERSONALIT
85 PR.""ROB THE ROBOT' OR YOU CAN 'LIVE' WITH HIM AS HE IS.
90 PR." WHEN FIRST BUILT HE WAS QUITE A 'BLAH' CHARACTER, B
95 PR."WITH YOUR HELP (AND OTHERS' TOO) ROB CAN TAKE ON A M
100 PR."COMPLEX PERSONALITY."
105 PR." IN THE PROGRAM YOU CAN CHOOSE TO:"
110 PR." 1) 'LIVE' WITH ROB AS HE IS;"
115 PR." 2) CHANGE ROB'S PERSONALITY IMMEDIATELY;"
120 PR." 3) CHANGE ROB'S PERSONALITY AFTER ":N4:" MORE US
125 PR." CHOOSE THIS OPTION."
130 PR." INPUT THE NUMBER OF YOUR CHOICE: 1,2,OR 3";
135 INPUT O1
140 OPEN /DATA/ FOR INPUT AS 2
145 INPUT FROM 2, N1
150 INPUT FROM 2, A(I) FOR I=1 TO N1
155 PR. "ROB HAS SEVERAL MOODS:"
160 PRINT I;A(I) FOR I=1 TO N1
165 INPUT FROM 2, N2
170 INPUT FROM 2, B(I) FOR I=1 TO N2
175 PR.PR. "THE 'TREATMENTS' THAT CAN BE GIVEN TO ROB ARE:
SOMEONE.....HIM"
180 PRINT I; B(I) FOR I=1 TO N2
185 INPUT FROM 2, N3
190 INPUT FROM 2, C(I) FOR I=1 TO N3
195 PR.PR. "ROB'S REACTION TO THE ABOVE 'TREATMENTS' CAN BE
HE..."
200 PRINT I;C(I) FOR I=1 TO N3
205 CLOSE 2

```

```

2100PEN /Q-W-MAT/ RANDOM INPUT 2
215 L=6
220FOR I=1 TO N1
225INPUT FROM 2 AT L: Q(I,J) FOR J=1 TO N2
230NEXT I
235FOR I=1 TO N1
240INPUT FROM 2 AT L: W(I,J) FOR J=1 TO N2
245NEXT I
250CLOSE 2
255 PR.PR." FIRST YOU'LL BE TOLD THE ROBOT'S 'MOOD' (HE WAS
260 PR."IN THIS 'MOOD' BY THE LAST USER). YOU SHOULD THEN
265 PR."ONE OF THE 'TREATMENTS' LISTED ABOVE."
270 IF O1=1 GOTO 340
275 PR."AT THAT POINT YOU WILL BE PROMPTED TO INPUT A 'REAC
280 PR."AND A 'MOOD' FROM THE LISTS ABOVE;"
285 PR."THESE ARE WHAT YOU EXPECT THE ROBOT'S RESPONSE WILL
290 PR."THE COMPUTER WILL RESPOND BY PRINTING 'AGREE' OR 'D
295 PR."FOLLOWED BY ROB'S PRESENTLY 'PROGRAMMED' 'REACTION'
300 PR."NEXT 'MOOD'."
305 PR." AGAIN YOU'LL BE PROMPTED TO INPUT A 'TREATMENT' TO
310 PR."ROBOT IN HIS NEW 'MOOD' AND THE CYCLE REPEATS."
315 IF O1=2 GOTO 340
320 PR."AFTER 10 INPUTS YOUR EXPECTED RESPONSES WILL BE REC
325 PR."ON A FILE. THEY WILL BE USED TO MODIFY THE ROBOT'S"
330 PR."PERSONALITY SO THAT HE WILL BECOME MORE AND MORE TH
335 PR."CHARACTER THAT YOU (AND OTHERS) SUGGEST."
340 DS= LEFT(A(M1),1)
345 YS= ",A,E,I,O,U,"
350 IF IEQIV(DS,YS) THEN LET DS="AN "+A(M1) ELSE LET DS="A
355 PR.PR." LET'S START...THE LAST USER LEFT ROB IN ":DS:"
360 PR."NOW TYPE IN A 'TREATMENT'..."
365T=1 !TALLY FOR NO. OF INPUTS
370 GOTO 385
375 T=T+1
380 PR.PR. "TREATMENT...":
385INPUT ES
390FOR I=1 TO N2
395 IF B(I) =ES LET M2=I, GOTO 415
400NEXT I
405PR."THERE'S SOME MISUNDERSTANDING OR MISTYPING."
410GOTO 380
415L1=Q(M1,M2) INEXT STATE
420I=W(M1,M2) IRESPONSE
425 IF O1=1 GOTO 455
430 GOSUB 2000
435 IF O1=3 GOTO 445
440 LET Q(M1,M2)=S(T) LET W(M1,M2)=R(T)
445IF L1=S(T) AND I=R(T) THEN PR."AGREE."; ELSE PR."DISAGRE
450 PR." ROB'S 'PROGRAMMED' PERSONALITY INDICATES THAT"

```

```

45500 NUM(3) GOTO 460,475,490
460 PR."OUR ":A(M1):" ROBOT ":C(I):" AND BECOMES"
465 PR.A(L1):" WHEN SOMEONE ":B(M2):" HIM."
470 GOTO 500
475 PR."WHEN SOMEONE ":B(M2):" ":A(M1):" ROB, HE"
480 PR. C(I):" AND BECOMES ":A(L1)
485 GOTO 500
490 PR."IF SOMEONE ":B(M2):" ROB WHILE HE IS"
495 PR.A(M1):" HE ":C(I):" AND BECOMES ":A(L1)
500 U(T)=M1 LET V(T)=M2
505 M1=L1
510 IF T<10 GOTO 375
515 L2=L1
520 IF O1=3 GOTO 545
525 IF O1=1 GOTO 730
530 PR.PR."LET'S SEE WHAT THE 'REFORMED' ROB IS LIKE."
535 PR."ROB IS NOW ":A(L2)
540 O1=1 LET T=0 GOTO 375
545 PR.PR."PLEASE WAIT UNTIL THE COMPUTER PRINTS A 'GOOD-BY
550 OPEN /RECORD/ RANDOM INPUT 8
555 LET L=0
560 INPUT FROM 8 AT L:L3
565 CLOSE 8
570 FOR T=1 TO 10
575 OPEN /RECORD/ RANDOM INPUT 8
580 L=5*N2*(U(T)-1) + 5*(V(T)-1)+3
585 L1=L+2
590 INPUT FROM 8 AT L:S1
595 INPUT FROM 8 AT L1:S2
600 IF S1>0 GOTO 635
605 CLOSE 8
610 OPEN /RECORD/ RANDOM OUTPUT 9
615 L=L-1
620 PRINT ON 9 AT L:R(T),S(T)
625 CLOSE 9
630 GOTO 695
635 IF S2=0 GOTO 655
640 L1=S2+4
645 INPUT FROM 8 AT L1:S2
650 GOTO 635
655 CLOSE 8
660 OPEN /RECORD/ RANDOM OUTPUT 9
665 L1=L1-1
670 PRINT ON 9 AT L1:L3
675 L1=L3
680 PRINT ON 9 AT L1:U(T),V(T),R(T),S(T)
685 L3=L3+5
690 CLOSE 9
695 IF T=5 PR."PLEASE WAIT"
700 NEXT T
705 OPEN /RECORD/ RANDOM OUTPUT 9
710 L=0
715 PRINT ON 9 AT L:L3
720 CLOSE 9

```

```
725 Z=Z+1
730 PR. PR. "YOU'VE LEFT ROB ":A(L2)
735 L=0
740 OPEN /Q-W-MAT/ RANDOM OUTPUT 3
745 PRINT ON 3 AT L:L2,Z
750 CLOSE 3
755 IF Z=N GOTO 775
760 PR.PR."LOGON AGAIN AT A LATER DATE TO SEE IF ROB HAS CI
765 PR."GOOD-BYE!"
770 STOP
775 LOAD "/MODIFIER/"
780 END
2000 PR. "EXPECTED REACTION...":
2005INPUT D$
2010 PR. "EXPECTED MOOD...":
2015INPUT E$
2020FOR K=1 TO N3
2025 IF C(K) =D$ LET R(T)=K,GOTO 2045
2030NEXT K
2035PR." NO SUCH RESPONSE"
2040GOTO 2000
2045FOR K=1 TO N1
2050 IF A(K)=E$ LET S(T)=K,GOTO 2065
2055NEXT K
2060GOTO 2035
2065 RETURN
```